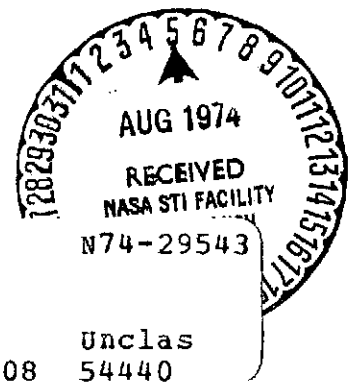
2mi4

NASA TECHNICAL MEMORANDUM

NASA TM X-71581

NASA TM X-71581



(NASA-TM-X-71581) COMPUTER PROGRAMS FOR CALCULATION OF MATRIX STABILITY AND FREQUENCY RESPONSE FROM A STATE-SPACE SYSTEM DESCRIPTION (NASA) 23 p HC $3.00
N74-29543
Unclas
CSCL 09B G3/08 54440

COMPUTER PROGRAMS FOR CALCULATION OF MATRIX STABILITY AND FREQUENCY RESPONSE FROM A STATE-SPACE SYSTEM DESCRIPTION

by Robert C. Seidel
Lewis Research Center
Cleveland, Ohio 44135
July 1974

ERRATA

NASA Technical Memorandum X-71581

COMPUTER PROGRAMS FOR CALCULATION OF MATRIX STABILITY AND FREQUENCY RESPONSE FROM A STATE-SPACE SYSTEM DESCRIPTION

by Robert C. Seidel

July 1974

Page 9: The statements in the DANSKY subroutine should read

```
      30 A (J,NMI) = SAVE                          (unchanged)
         DO 40 J = 1, NN                           (changed)

      90 SUM = SUM + A (NMIP1,K)*A(K,11)           (unchanged)
         IF(11.LT.NMI) GO TO 100                   (unchanged)
         IF(11.EQ.NN)  GO TO 100                   (unchanged)
         SUM = SUM + A(NMIP1,11+1)                 (unchanged)
     100 A(NMI,11) = SUM                           (unchanged but three
                                                   duplicate lines removed)
```

Page 14: The statements should read

CALL BOLLIN (A,AS,B,C,X,Y,Z1,Z2,Z3,N,NMAX)

DOUBLE PRECISION: AS, X,Y,Z1,Z2,Z3 must be double precision

NOTES: (1) A and B may be destroyed in call to DAVISO

This information is being published in preliminary form in order to expedite its early release.

ABSTRACT

FORTRAN computer subroutines stemming from requirements to process state variable system equations for systems of high order are presented. They find the characteristic equation of a matrix using the method of Danilevsky, the number of roots with positive real parts using the Routh-Horwitz alternate formulation, convert a state variable system description to a Laplace transfer function using the method of Bollinger, and evaluate that transfer function and obtain its frequency response. A sample problem is presented to demonstrate use of the subroutines.

E-8016

COMPUTER PROGRAMS FOR CALCULATION OF MATRIX STABILITY AND FREQUENCY RESPONSE FROM A STATE-SPACE SYSTEM DESCRIPTION

by Robert C. Seidel

Lewis Research Center

SUMMARY

FORTRAN computer subroutines stemming from requirements to process state variable system equations for systems of high order are presented. They find the characteristic equation of a matrix using the method of Danilevsky, the number of roots with positive real parts using the Routh-Horwitz alternate formulation, convert a state variable system description to a Laplace transfer function using the method of Bollinger, and evaluate that transfer function and obtain its frequency response. A sample problem is presented to demonstrate use of the subroutines.

E-8016

INTRODUCTION

High-speed digital computers have made matrix state variable methods for system analysis practical. But for large systems, the required execution time and the cumulative effect of round off errors make it increasingly important to employ efficient algorithms. FORTRAN subroutines resulting from requirements to handle large systems are reported herein. In particular, they find the characteristic equation of a system matrix, test that equation for the number of roots with positive real parts, convert a state variable system description to a Laplace transfer function and evaluate the transfer function at a given frequency to obtain its frequency response.

The FORTRAN program for obtaining the characteristic equation of a system matrix uses the method of Danilevsky, reference 1. The program includes a Gauss pivotal element condensation scheme to somewhat increase its accuracy. A competing method is that of Leverrier, references 1 and 2. However, to compute the characteristic polynomial the Danilevsky method is known to be more accurate, faster, and require less storage. The FORTRAN program for determining the stability of the characteristic polynomial uses the Routh-Horwitz alternate formulation method, reference 3. The FORTRAN program for obtaining the system transfer function uses the method of Bollinger, reference 4. This method appears more reliable than the method of Davison, reference 5, which in certain cases is known to converge improperly (ref. 4). However Davison's transformation, which permits the output to be an arbitrary linear combination of the states, is used in

conjunction with Bollinger's method. The inverse of this transformation is also required, but is easily calculated in closed form. Once the transfer function is obtained its frequency response is easily calculated.

The FORTRAN listings and subroutine descriptions are presented in the following section. Equation symbols are defined in Appendix A. FORTRAN symbols are defined separately for each program. A sample problem demonstrating the use of the subroutines is described in Appendix B.

COMPUTER PROGRAMS

Characteristic Equation

In subroutine DANSKY the characteristic equation of a matrix is found by the method of Danilevsky, reference 1. The characteristic equation of an nxn matrix A is an expansion of the determinant equation

$$|A - \lambda I| = (-1)^n(\lambda^n + Z1_1\lambda^{n-1} + \cdots + Z1_n) = 0$$

where I is the identity matrix and the polynomial coefficients sought are in the Z1 vector. The Z1 vector coefficients are obtained in DANSKY through successive application of similarity transformations which finally produce the Z1 vector in the top row of A. As noted in reference 1 the method allows use of a Gauss pivotal element scheme to somewhat increase its accuracy. In DANSKY this option is implemented. The Gauss method performs similarity transformations which interchange columns and rows of A to place the element with the largest absolute value in pivot position. The execution time for an 18x18 A matrix is about 0.35 sec (IBM-360-67 TSS computer). DANSKY uses (as most of the programs) double precision. The time penalty for using double precision is only about 0.02 sec for the 18x18 matrix. One problem which occurs with certain A matrices when using DANSKY is that of exponent under or overflow. In such cases the matrix A may be (time) scaled by multiplying each element of A by a positive constant, r. The characteristic polynomial of the scaled matrix becomes

$$\lambda^n + (Z1_1 r)\lambda^{n-1} + \cdots + (Z1_n r^n)$$

Figure 1 is a description of the DANSKY calling statement transfer variables. Figure 2 is a FORTRAN listing of DANSKY.

For comparison with the Danilevsky method some results obtained with the Leverrier method are cited. For the same 18x18 matrix discussed earlier, the Leverrier program required 4.2 sec (compared to 0.35 sec for the Danilevsky method). Also, two additional 18x18 double precision scratch storage matrices are required for the Leverrier method, and double precision is required to obtain the same accuracy achieved by DANSKY in single precision.

In DANSKY a call is made to subroutine POLMPY to multiply two polynomials. In certain cases the method of Danilevsky obtains the characteristic equation coefficients in partially factored form and the factors must be multiplied together to obtain the characteristic equation. In DANSKY a call is made to POLMPY in all cases with one of the polynomials possibly unity. Figure 3 is a description of the POLMPY transfer variables. Figure 4 is a FORTRAN listing of POLMPY.

Stability

The subroutine RHWTZ performs a stability test upon the characteristic equation. It counts the number of roots with positive real parts without actually finding them. A simple recursive algorithm which is well suited to machine computation is used. Its description is given in reference 3. The program is specialized in that it assumes the leading polynomial coefficient is unity as in the form returned by DANSKY. The execution time for an eighteenth order polynomial is about 0.007 sec. Figure 5 presents a description of the RHWTZ transfer variables and figure 6 presents a FORTRAN listing of RHWTZ.

For certain equations the test may fail if during the algorithm execution a zero appears as a divisor term. In this case the number of unstable roots is set to -1 and the message "Test Failed M set to -1" is output.

Transfer Function

BOLLIN is a subroutine for converting a state variable matrix differential equation into an equivalent Laplace transfer function. The system equations considered are

$$\dot{x} = Ax + Bu, \; y = C^t x$$

where x, B, and C are n vectors, A is an nxn matrix, and u and y are input and output scalars. The following steps are taken to obtain the system transfer function:

1. A call to subroutine DAVISO transforms the A and B system matrices using the C vector so as to make the output y a state variable in the transformed system, reference 5. The transformed system is

$$A^* = TAT^{-1},\ B^* = TB,\ \text{and}\ x^* = Tx$$

such that

$$\dot{x}^* = A^*x^* + B^*u$$

where T is the identity matrix except that the MC-th row is overwritten by the C^t vector. The integer MC is the position of the element of C with the maximum absolute value. The T^{-1} matrix is similar to inverses encountered in the proof of the Danilevsky method, reference 1, and can be written down explicitly. The output y^* of the modified system is the MC-th modified state variable, x^*_{MC}.

2. The denominator polynomial Z1 (characteristic equation) of the system transfer function is obtained by a call to DANSKY using the A^* matrix.

3. The numerator polynomial is obtained in two more calls to DANSKY using Bollinger's method, reference 4. First, with the MC-th column of A^* replaced by $-B^*$ to obtain Z2; then, with a matrix of order (n-1) obtained by deleting the MC-th row and column from A^* to obtain Z3.

4. The numerator polynomial is computed as Z4 = Z2 - sZ3.

5. The system transfer function y(s)/u(s) is Z4(s)/Z1(s).

Execution times found for various order systems were about 1.0 sec for an eighteenth order, 2.7 sec for a twenty-sixth order, and 9.7 sec for a forty-first order system. Figure 7 is a description of the BOLLIN transfer variables. Figure 8 is a FORTRAN listing of BOLLIN. Figure 9 is a description of the DAVISO transfer variables and figure 10 is a FORTRAN listing of DAVISO. If the A matrix is time scaled by multiplication by scalar r, the B vector and frequencies used to evaluate the transfer function should also be scaled by r. To handle the more general case of multiple input, multiple output systems where B and C are matrices, the program calling BOLLIN would start the B and C matrices at the appropriate columns in the transfer variable list to obtain the desired input/output relation, reference 5.

Frequency Response

The subroutine FRPOLY may be used to evaluate the Laplace transform Z4(s)/Z1(s) ratio of polynomials for a given frequency $s = j\omega$. The evaluation is performed in double precision with the real and imaginary parts of the powers of $j\omega$ handled separately. Figure 11 is a description of the FRPOLY transfer variables and figure 12 is a FORTRAN listing of FRPOLY. To evaluate a system frequency response over a range of frequencies, multiple calls to FRPOLY would be made.

CONCLUDING REMARKS

Six subroutines: DANSKY, POLMPY, RHWTZ, BOLLIN, DAVISO, and FRPOLY were presented and discussed. The chart shown in figure 13 summarizes the flow from a state variable representation of a system through the various subroutines. The major input and output relations and alternate paths for various uses are noted.

REFERENCES

1. Faddeeva, V. N.: Computational Methods of Linear Algebra. Dover Publications, 1959, pp. 166-176.

2. Zadeh, Lotfi A., and Desoer, Charles A.: Linear System Theory; The State Space Approach. McGraw-Hill Book Co., Inc., 1963, pp. 303-305.

3. Korn, Gravins A., and Korn, Theresa M.: Mathematical Handbook for Scientists and Engineers. 2nd ed., McGraw-Hill Book Co., Inc., 1961.

4. Bollinger, K. E., and Mathur, J. C.: To Compute the Zeros of Large Systems. IEEE Trans., vol. AC-16, no. 1, Feb. 1971, pp. 95-96.

5. Davison, E. J.: A Computational Method for Finding the Zeros of a Multivariable Linear Time Invariant System. Automatica, vol. 6, no. 3, May 1970, pp. 481-484.

APPENDIX A

SYMBOLS

A system matrix, nxn

B input vector, n

C output vector, n

I identity matrix, nxn

j imaginary, $\sqrt{-1}$

MC position of element in C with maximum absolute value

n system order

r scaling factor

s Laplace variable, sec^{-1}

t time, sec

T transformation matrix, nxn

u input

x state variable vector, n

y output

Z1 characteristic equation polynomial

Z2 intermediate polynomial

Z3 intermediate polynomial

Z4 system numerator transfer function polynomial

λ root of characteristic equation

ω frequency, hertz

Superscript:

* denotes transformed variable

t transpose

APPENDIX B

SAMPLE PROBLEM

A third order sample problem demonstrating the combined use of the subroutines is described next. The problem studied is the transfer function

$$\frac{y(s)}{u(s)} = \frac{s + 6}{(s^2 + 3s + 9)(s + 4)} = \frac{s + 6}{s^3 + 7s^2 + 21s + 36}$$

In phase variable form the state matrices are

$$A = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ -36 & -21 & -7 \end{bmatrix}, \quad B = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}, \quad C = \begin{bmatrix} 6 \\ 1 \\ 0 \end{bmatrix}$$

Figure 14 is a FORTRAN listing of the sample problem MAIN program. First the A, B, and C matrices are output. Next after a call to BOLLIN, the denominator and numerator polynomial coefficients are output. Then, after a call to FRPOLY, the transfer function evaluated at one hertz is output, and finally, after a call to RHWTZ, the number of unstable roots is output.

Figure 15 is a listing of the program output. The denominator polynomial is $s^3 + 7s^2 + 21s + 36$ with the unity coefficient of s^3 understood. The numerator polynomial is $-2.220E{-}16\ s^2 + s + 6$. The coefficient of the s^2 term should actually be zero but due to limited numerical precision is slightly in error. The transfer function is evaluated at one hertz ($s = j2\pi$) and has a real part of $-.03048$ and an imaginary part of $-.01142$ or an amplitude of $.03255$ at -159.5 degrees. There are no roots of the denominator polynomial with positive real parts.

If only a test of system stability were desired then it would not be necessary to use BOLLIN. Instead DANSKY could be called to obtain the characteristic equation followed by a call to RHWTZ. BOLLIN is organized such that it computes the denominator transfer function Z1 each time it is called. To save computations the user may wish to modify this if the transformed matrix A^* is known to remain constant for a particular set of B and/or C changes.

DANSKY

Danilevsky

PURPOSE: To obtain the characteristic equation of a square real matrix

USAGE: CALL DANSKY (A, X, Y, Z, N, NMAX)

Where A - square real matrix, order NxN

X - scratch vector, order N

Y - scratch vector, order N

Z - characteristic polynomial vector, order N

N - order of A

NMAX - dimension of A in calling program $\geq$ N

DOUBLE DIMENSION: A must be double dimensioned (NMAX, NMAX) in calling program

DOUBLE PRECISION: A, X, Y, and Z must be double precision in calling program

NOTES: (1) A is destroyed in obtaining Z

(2) The Z characteristic equation is returned in the form

$$\lambda^n + Z_1\lambda^{n-1} + \cdots + Z_{n-1}\lambda + Z_n$$

Subroutines called: (1) POLMPY

FIGURE 1.-DESCRIPTION OF SUBROUTINE DANSKY TRANSFER VARIABLES

```
C  COMPUTE THE COEFFICIENTS OF THE CHARACTERISTIC EQUATION
      SUBROUTINE DANSKY(A,X,Y,Z,N,NMAX)
      DIMENSION A(NMAX,1),X(1),Z(1),Y(1)
      DOUBLE PRECISION SUM,A,SAVE,PIVOT,CK,X,Y,Z
      NN=N
10    NM1=NN-1
      IF(NM1.EQ.0) GO TO 125
      DO 120 I=1,NM1
      NMI=NN-I
      IPVT=NMI
      NMIM1=NMI-1
      NMIP1=NMI+1
C  FIND MAXIMUM ELEMENT IN PIVOT ROW
      XMAX=A(NMIP1,NMI)
      IF(NMIM1.GT.0) GO TO 15
      IF(XMAX.EQ.0.) GO TO 140
      GO TO 50
15    DO 20 K1=1,NMIM1
      K=NMI-K1
      IF(ABS(SNGL(A(NMIP1,K))).LE.ABS(XMAX)) GO TO 20
      XMAX=A(NMIP1,K)
      IPVT=K
20    CONTINUE
      IF(XMAX.EQ.0.) GO TO 140
      IF(IPVT.EQ.NMI) GO TO 50
C  SIMULARITY TRANSFORM SO PIVOT ELEMENT IS THE MAXIMUM
      DO 30 J=1,NMIP1
      SAVE=A(J,IPVT)
      A(J,IPVT)=A(J,NMI)
30    A(J,NMI)=SAVE
      DO 40 JJ1,NN
      SAVE=A(IPVT,J)
      A(IPVT,J)=A(NMI,J)
40    A(NMI,J)=SAVE
C  A UPDATE EQUALS A * M(SUB N-1)
50    PIVOT=1./A(NMIP1,NMI)
      DO 80 K=1,NN
      IF(K.EQ.NMI) GO TO 80
      CK=-A(NMIP1,K)*PIVOT
      DO 70 II=1,NMI
70    A(II,K)=A(II,K)+A(II,NMI)*CK
80    CONTINUE
      DO 85 II=1,NMI
85    A(II,NMI)=A(II,NMI)*PIVOT
C  MULT M(SUB N-1) INVERSE TIMES A UPDATED
      DO 110 II=1,NN
      SUM=0.
      DO 90 K=1,NMI
90    SUM=SUM+A(NMIP1,K)*A(K,II)
      IF(II.LT.NMI) GO TO 100
      IF(II.EQ.NN) GO TO 100
      SUM=SUM+A(NMIP1,II+1)
      IF(II.LT.NMI) GO TO 100
      IF(II.EQ.NN) GO TO 100
      SUM=SUM+A(NMIP1,II+1)
100   A(NMI,II)=SUM
110   CONTINUE
120   CONTINUE
125   NMIP1=1
C  ELEMENTS OF C SET EQUAL TO LAST ELEMENTS IN ROW NMIP1 OF =A
140   DO 150 J=NMIP1,NN
      NX=J-NMIP1+1
150   X(NX)=-A(NMIP1,J)
C  MULTIPLY OUT FACTORS OF CHARACTERISTIC EQUATION
190   NY=N-NN
      DO 200 J=1,NY
200   Y(J)=Z(J)
      CALL POLMPY(X,Y,Z,NX,NY,NYPNX)
C  REDUCE SYSTEM ORDER
      NN=NN-NX
      IF(NN.GT.0) GO TO 10
      RETURN
      END
```

FIGURE 2.-FORTRAN LISTING FOR SUBROUTINE DANSKY

POLMPY

Polynomial Multiplication

PURPOSE: To multiply two polynomials X*Y

CALL POLMPY (X, Y, Z, NX, NY, NZ)

Where X - first polynomial, vector

Y - second polynomial, vector

Z - returned product polynomial, vector

NX - order of X

NY - order of Y

NZ = NX + NY (order of Z) returned

NOTE: (1) leading coefficient of highest power is assumed to be one, i.e., for X:

$$\lambda^{NX} + X_1\lambda^{NX-1} + \cdots + X_{NX}$$

DOUBLE PRECISION: X, Y, and Z must be double precision in the calling program

FIGURE 3.-DESCRIPTION OF SUBROUTINE POLMPY TRANSFER VARIABLES

```
C  MULTIPLY X*Y=Z (LEADING POLYNOMIAL COEFFICIENTS ASSUMED UNITY)
      SUBROUTINE POLMPY(X,Y,Z,NX,NY,NZ)
      DIMENSION X(1),Y(1),Z(1)
      DOUBLE PRECISION X,Y,Z,YJ
      NZ=NX+NY
C  IF NX=0 MEANS POLY X=1;THEREFORE Z=Y
      IF(NX.GT.0) GO TO 6
      DO 4 J=1,NY
4     Z(J)=Y(J)
      GO TO 40
C  IF NY=0 MEANS POLY Y=1;THEREFORE Z=X
6     IF(NY.GT.0) GO TO 15
      DO 8 J=1,NX
8     Z(J)=X(J)
      GO TO 40
C  START MULTIPLICATION BY MAKING Z=Y*S**NX
15    DO 20 J=1,NZ
      YJ=Y(J)
      IF(J.GT.NY) YJ=0.
20    Z(J)=YJ
C  MULTIPLICATION LOOP
      DO 30 K=1,NX
      Z(K)=Z(K)+X(K)
      DO 25 J=1,NY
      KPJ=K+J
25    Z(KPJ)=Z(KPJ)+Y(J)*X(K)
30    CONTINUE
40    RETURN
      END
```

FIGURE 4.-FORTRAN LISTING FOR SUBROUTINE POLMPY

RHWTZ

ROUTH - HURWITZ ALTERNATE FORMULATION

PURPOSE: To compute the number of roots with positive real parts of a polynomial equation

USAGE: CALL RHWTZ (C,N,M)

Where C - coefficients of polynomial equation, vector

N - equation order

M - number of roots with positive real parts (M set to -1 if test fails due to attempted division by zero)

RESTRICTION: The C vector starts with the second coefficient, as the first is assumed unity; that is:

$$\lambda^n + C_1\lambda^{n-1} + \cdots + C_{n-1}\lambda + C_n = 0$$

NOTE: C is destroyed upon return

DOUBLE PRECISION: C must be double precision in the calling program

PROGRAM OUTPUT: "TEST FAILED M SET TO -1" output if division by zero is attempted

FIGURE 5.-DESCRIPTION OF SUBROUTINE RHWTZ TRANSFER VARIABLES

```
C  ROUTH HORWITZ ALTERNATE FORMULATION STABILITY TEST
      SUBROUTINE RHWTZ(C,N,M)
      DOUBLE PRECISION C,CNM1,COEF,CNM2
      DIMENSION C(1)
      NM1=N-1
      M=0
C  PROVIDE FOR CASE N=0,1,OR 2
      IF(N.LE.0) GO TO 30
      IF(N-2) 4,8,10
4     IF(C(1).LT.0.) M=1
      GO TO 30
8     CNM1=1.
      GO TO 25
10    IF(C(1).EQ.0.) GO TO 35
      COEF=1./C(1)
C  START ALGORITHM LOOP
      DO 20 K=2,NM1
      IF(COEF.LT.0) M=M+1
      DO 15 J=K,NM1,2
15    C(J)=C(J)-COEF*C(J+1)
      IF(C(K).EQ.0.) GO TO 35
      COEF=C(K-1)/C(K)
20    CONTINUE
C  FINISH REMAINING 2 ND ORDER POLYNOMIAL
      CNM2=C(N-2)
25    IF(CNM2*C(NM1).LT.0.) M=M+1
23    IF(C(NM1)*C(N).LT.0.) M=M+1
30    RETURN
35    WRITE(6,45)
45    FORMAT(1X,23HTEST FAILED M SET TO -1)
      M=-1
      GO TO 30
      END
```

FIGURE 6.-FORTRAN LISTING FOR SUBROUTINE RHWTZ

BOLLIN

Bollinger

PURPOSE: To obtain a system transfer function from system matrix equations

USAGE: CALL BOLLIN (A,AS,B,X,Y,Z1,Z2,Z3,N,NMAX)

Where A - system matrix (NxN)

AS - scratch matrix (NxN)

B - system input vector (N)

C - system output vector (N)

X - scratch vector (N)

Y - scratch vector (N)

Z1 - transfer function denominator, vector (N)

Z2 - transfer function numerator, vector (N)

Z3 - scratch, vector (N)

N - system order

NMAX - dimension of A and AS in calling program $\geq$ N

DOUBLE DIMENSION: A and AS must be double dimensioned (NMAX, NMAX) in calling program

DOUBLE PRECISION: AS must be double precision in calling program

NOTES: (1) A is destroyed in obtaining Z1 and Z2

(2) The Z1 and Z2 are returned as in

$$Z1 = \lambda^n + Z1_1\lambda^{n-1} + \cdots + Z1_n$$

$$Z2 = \quad Z2_1\lambda^{n-1} + \cdots + Z2_n$$

SUBROUTINES CALLED: (1) DAVISO

(2) DANSKY

FIGURE 7.-DESCRIPTION OF SUBROUTINE BOLLIN TRANSFER VARIABLES

```
C  CONVERT X(DOT)=AX+BU,Y=C(TRANSPOSE)X TO TRANSFER
C  FUNCTION Y/U=Z2/Z1 RATIO OF POLYNOMIALS
      SUBROUTINE BOLLIN(A,AS,B,C,X,Y,Z1,Z2,Z3,N,NMAX)
      DIMENSION A(NMAX,1),AS(NMAX,1),B(1),C(1),X(1),Y(1)
      DIMENSION Z1(1),Z2(1),Z3(1)
      DOUBLE PRECISION AS,X,Y,Z1,Z2,Z3
C  TRANSFORM Z=TX TO MAKE OUTPUT A STATE
      CALL DAVISO(A,B,C,N,NMAX,MC)
C  SAVE A USING AS
      DO 20 K=1,N
      DO 10 J=1,N
10    AS(K,J)=A(K,J)
20    CONTINUE
C  FIND DEN CHAR EQN COEF'S
      CALL DANSKY(AS,X,Y,Z1,N,NMAX)
C  SET AS=A AGAIN AND OVER WRITE A(K,MC) COLUMN WITH -B
      DO 40 K=1,N
      DO 30 J=1,N
30    AS(K,J)=A(K,J)
40    AS(K,MC)=-B(K)
C  FIND FIRST PART OF NUM CHAR EQN
      CALL DANSKY(AS,X,Y,Z2,N,NMAX)
C  SET AS=A AGAIN; COLLAPSE MC ROW AND COLUMN
      NM1=N-1
      DO 60 K=1,NM1
      K1=K
      IF(K.GE.MC) K1=K+1
      DO 50 J=1,NM1
      J1=J
      IF(J.GE.MC) J1=J+1
50    AS(K,J)=A(K1,J1)
60    CONTIQUE
C  FIND SECOND PART OF NUM CHAR EQN
      CALL DANSKY(AS,X,Y,Z3,NM1,NMAX)
C  SUBSTRACT FIRST PART FROM SECOND PART
      DO 100 J=1,NM1
100   Z2(J)=Z2(J)-Z3(J)
      RETURN
      END
```

FIGURE 8.-FORTRAN LISTING FOR SUBROUTINE BOLLIN

DAVISO

Davison

PURPOSE: Transform system to make output a state variable

USAGE: CALL DAVISO (A,B,C,N,NMAX,MC)

Where A - input system matrix which upon return is transformed system TAT^{-1}

B - input system vector which upon return is transformed input vector TB

C - system output vector

N - order of A

NMAX - dimension of A in calling program $\geq$ N

MC - position of maximum element in C vector

DOUBLE DIMENSION: A must be double dimensioned (NMAX, NMAX) in calling program

PROGRAM OUTPUT: If all N elements of C are zero the message "All elements of C are zero" is output and the program is put in PAUSE

FIGURE 9.-DESCRIPTION OF SUBROUTINE DAVISO TRANSFER VARIABLES

```
C  TRANSFORM X(DOT)=AX+BU,Y=C(TRANSPOSE)X USING Z=TX SUCH
C  THAT Y IS A STATE VARIABLE OF Z(DOT)=TAT(INVERSE)+TBU
      SUBROUTINE DAVISO(A,B,C,N,NMAX,MC)
      DIMENSION A(NMAX,1),B(1),C(1)
      DOUBLE PRECISION SUM
C  FIND MAX ELEMENT IN C AS C(MC)
      CM=0.
      MC=0
      DO 5 J=1,N
      IF(ABS(C(J)).LT.CM) GO TO 5
      CM=ABS(C(J))
      MC=J
5     CONTINUE
      IF(MC.GT.0) GO TO 10
      WRITE(6,7)
7     FORMAT(1X,26HALL ELEMENTS OF C ARE ZERO)
      PAUSE
C  PREMULT A BY T CHANGES MC ROW ONLY
10    DO 17 K=1,N
      SUM=0.
      DO 15 J=1,N
15    SUM=SUM+DBLE(C(J)*A(J,K))
17    A(MC,K)=SUM
C  POST MULT A BY T INVERSE
C             FOR J NOT = MC: A(K,J)=A(K,J)-A(K,MC)*C(J)/C(MC)
      PIVOT=1./C(MC)
      DO 30 J=1,N
      IF(J.EQ.MC) GO TO 30
      DO 25 K=1,N
25    A(K,J)=A(K,J)-A(K,MC)*C(J)*PIVOT
30    CONTINUE
C             FOR J=MC:       A(K,MC)=A(K,MC)/C(MC)
      DO 35 K=1,N
35    A(K,MC)=A(K,MC)*PIVOT
C  PRE MULT B BY T CHANGES B(MC) ONLY
      SUM=0.
      DO 37 J=1,N
37    SUM=SUM+DBLE(C(J)*B(J))
      B(MC)=SUM
      RETURN
      END
```

FIGURE 10.-FORTRAN LISTING FOR SUBROUTINE DAVISO

FRPOLY

Frequency Response of Polynomials

PURPOSE: To evaluate system transfer function polynomials to obtain system frequency response

USAGE: CALL FRPOLY (Z1,Z2,HZ,G,AMP,PHA,N)

Where

Z1 - denominator polynomial, vector order N

Z2 - numerator polynomial, vector order N

HZ - frequency in hertz

G - system transfer function Z2(jHZ)/Z1(jHZ) evaluated at radian frequency HZ*2π

AMP - transfer function amplitude, $|G|$

PHA - transfer function phase, $\angle G$ degrees

N - order of Z1

COMPLEX: G must be declared complex in the calling program

DOUBLE PRECISION: Z1 and Z2 must be declared double precision in the calling program

NOTES: the Z1 and Z2 must be in the form:

$$Z1 = \lambda^n + Z1_1\lambda^{n-1} + \cdots + Z1_n$$

$$Z2 = \quad Z2_1\lambda^{n-1} + \cdots + Z2_n$$

FIGURE 11.-DESCRIPTION OF SUBROUTINE FRPOLY TRANSFER VARIABLES

```
C  EVALUATE TRANSFER FUNCTION Z2(S)/Z1(S) FOR S=6.28*HZ*J
      SUBROUTINE FRPOLY(Z1,Z2,HZ,G,AMP,PHA,N)
      DIMENSION Z1(1),Z2(1)
      DOUBLE PRECISION GT,WJ,SUM1R,SUM1I,SUM2R,SUM2I,ZZ1,ZZ2,Z1,Z2
      COMPLEX G,SUM1,SUM2
      NM1=N-1
      SUM1R=Z1(N)
      SUM1I=0.
      SUM2R=Z2(N)
      SUM2I=0.
      GT=1.
      WJ=HZ*6.2831853
1     KK=2
      IF(NM1.EQ.0) GO TO 40
      DO 35 K=1,NM1
      NMK=N-K
      GT=GT*WJ
      ZZ1=Z1(NMK)*GT
      ZZ2=Z2(NMK)*GT
      GO TO (5,10,3,8),KK
3     ZZ1=-ZZ1
      ZZ2=-ZZ2
5     SUM1R=SUM1R+ZZ1
      SUM2R=SUM2R+ZZ2
      GO TO 30
8     ZZ1=-ZZ1
      ZZ2=-ZZ2
10    SUM1I=SUM1I+ZZ1
      SUM2I=SUM2I+ZZ2
30    KK=KK+1
      IF(KK.GT.4) KK=1
35    CONTINUE
C  ADD ON S**N TO DEN SUM1
40    SUM1=CMPLX(SNGL(SUM1R),SNGL(SUM1I))+GT*WJ*(CMPLX(0.,1.))**(KK-1)
      SUM2=CMPLX(SNGL(SUM2R),SNGL(SUM2I))
      G=SUM2/SUM1
      AMP=CABS(G)
      PHA=ATAN2(AIMAG(G),REAL(G))*57.29578
      RETURN
      END
```

FIGURE 12.-FORTRAN LISTING FOR SUBROUTINE FRPOLY

SYSTEM STATE VARIABLE EQUATIONS

$$\dot{x} = Ax + Bu, \quad y = C^t x$$

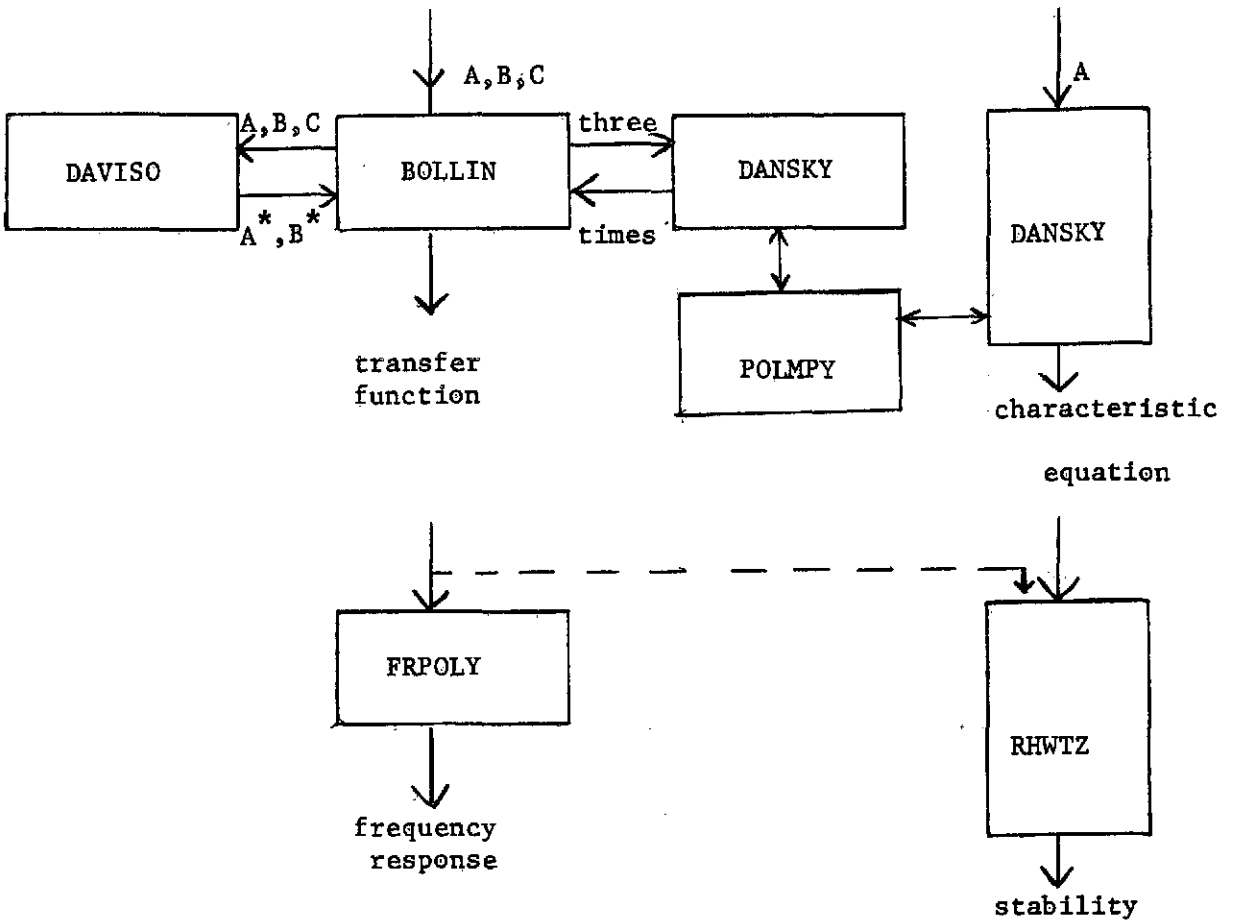


FIGURE 13.-FLOW CHART FOR FORTRAN SUBROUTINES

```
C  SAMPLE PROBLEM MAIN PROGRAM
      DIMENSION A(3,3),AS(3,3),B(3),C(3),Z1(3),Z2(3),Z3(3),X(3),Y(3)
      DOUBLE PRECISION AS,Z1,Z2,Z3,X,Y
      COMPLEX G
      DATA A/0.,0.,-36.,1.,0.,-21.,0.,1.,-7./
      DATA B/0.,0.,1./,C/6.,1.,0./,HZ,N,NMAX/1.,3,3/
      WRITE(6,1) ((A(I,J),J=1,N),I=1,N)
1     FORMAT(' A=',/,1P3E12.3,/,1P3E12.3,/,1P3E12.3)
      WRITE(6,2) (B(J),J=1,N)
2     FORMAT(' B VECTOR=',1P3E12.3)
      WRITE(6,3) (C(J),J=1,N)
3     FORMAT(' C VECTOR=',1P3E12.3)
      CALL BOLLIN(A,AS,B,C,X,Y,Z1,Z2,Z3,N,NMAX)
      WRITE(6,10) (Z1(J),J=1,N)
10    FORMAT(' DENOMINATOR=',1P3E12.3)
      WRITE(6,20) (Z2(J),J=1,N)
20    FORMAT(' NUMERATOR=',1P3E12.3)
      CALL FRPOLY(Z1,Z2,HZ,G,AMP,PHA,N)
      WRITE(6,30) HZ,G,AMP,PHA
30    FORMAT(' AT',F4.1,' HERTZ THE TRANSFER FUNCTION =',-
     1  1P2E12.3,/,' AMPLITUDE=',1PE12.3,' PHASE=',1PE12.3)
      CALL RHWTZ(Z1,N,M)
      WRITE(6,40) M
40    FORMAT(' NUMBER OF UNSTABLE ROOTS=',I5)
      STOP
      END
```

FIGURE 14.- FORTRAN listing for sample problem main program

```
A=
  0.000         1.000E 00    0.000
  0.000         0.000        1.000E 00
 -3.600E 01   -2.100E 01   -7.000E 00
B VECTOR=    0.000         0.000         1.000E 00
C VECTOR=    6.000E 00     1.000E 00     0.000
DENOMINATOR=    7.000E 00    2.100E 01    3.600E 01
NUMERATOR=   -2.220E-16     1.000E 00    6.000E 00
AT 1.0 HERTZ THE TRANSFER FUNCTION =  -3.048E-02  -1.142E-02
AMPLITUDE=    3.255E-02 PHASE=   -1.595E 02
NUMBER OF UNSTABLE ROOTS=     0
```

FIGURE 15.-SAMPLE PROBLEM PROGRAM OUTPUT